

2 - ISTRUZIONI DI INPUT-OUTPUT E DI ASSEGNAZIONE

INPUT-OUTPUT

write ('Carissima ti bacio ', x, ' volte')

- Scrivi la stringa "Carissima ti bacio "
- poi il valore della variabile x (NOTA)
- infine la stringa " volte"

NOTA: ossia, il contenuto che ha in quel momento la locazione di memoria ("scatoletta") x

writeln ('a+b = ', a+b)

- Scrivi la stringa "a+b = "
- poi il valore dell'espressione a+b (il valore della somma dei contenuti delle "scatolette" a, b)
- infine, **vai a capo** col cursore (writeln e non write)

read (num) Leggi ciò che l'utente digita sulla tastiera e mettilo nella "scatoletta" num

NOTA: l'utente, dopo aver digitato, dovrà premere il tasto "Invio" sulla tastiera

readln (num) Come prima; alla fine, però, vai a capo col cursore

NOTA: sovente ci permetteremo di scrivere, alla buona, "scatoletta" anziché "locazione di memoria"

ASSEGNAZIONE

Il simbolo dell' "assegnazione" in PASCAL *NON* è =, bensì è :=

num:=5 Assegna alla variabile num (cioè: metti nella scatoletta num) il valore 5

y:=a+b Assegna alla variabile y (cioè: metti nella scatoletta y) il valore dato dalla somma dei valori che in quel momento hanno le variabili a, b (cioè: che in quel momento stanno nelle scatolette a, b)

c:=c+1 Assegna alla variabile c (ossia: metti nella scatoletta c) il valore che c aveva **PRECEDENTEMENTE**, aumentato di 1 (quindi, in pratica: **incrementa di una unità il valore della variabile c**)
:= non indica uguaglianza, indica assegnazione!



ESEMPI

```
program moltiplicazioni_1;
uses crt;
var a, b: longint;
begin
```

```
  clrscr;
  readln (a);
  readln (b);
  writeln (a*b);
  readln;
```

```
end.
```

OSSERVAZIONI

a*b

Il simbolo di moltiplicazione non si può sottintendere all'interno di una espressione matematica in Pascal, e si realizza con un asterisco

readln

Istruzione indispensabile se si vuole avere tempo di osservare l'output: prova a toglierla, fai eseguire (=dai il "Run") e vedrai!

clrscr

Prova a toglierlo, dai il Run per 2 esecuzioni consecutive e vedi che succede ...

```
program moltiplicazioni_2;
uses crt;
(*questo programma è simile
al precedente, è solo più ricco*)
var a, b, prod: longint;
begin
```

```
  clrscr;
  write ('a = ');
  readln (a);
  write ('b = ');
  readln (b);
  prod:=a*b;
  write ('a*b = ');
  writeln (prod);
```

```
readln;
end.
```

OSSERVAZIONI

(* *)

Le "parentesi asteriscate" consentono di inserire un commento in un punto qualsiasi del programma

write ('a = ')

ha qui la funzione di mandare in output una scritta che aiuti l'utente a capire cosa deve fare.

```
program moltiplicazioni_3;
uses crt;
var a, b: longint;
```

```
begin
```

```
  clrscr;
  write ('a = ');
  readln (a);
  write ('b = ');
  readln (b);
```

```
  write ('a*b = ', a*b);
```

```
readln;
end.
```

OSSERVAZIONI

Organizzando il nostro programma in questo modo, possiamo evitare di introdurre la variabile "prod"

Domanda: se anziché scrivere

```
write ('a*b = ', a*b)
```

scrivessimo

```
write (a, '*', b, '=', a*b)
```

come cambierebbe l'output?

Ogni coppia ' ' di APICI serve a indicare che il computer dovrà scrivere sul monitor **ESATTAMENTE** quella **SEQUENZA DI CARATTERI (=STRINGA)** che è contenuta entro gli apici stessi. In **ASSENZA DI APICI**, va invece scritto il **VALORE di una variabile (o espressione)**